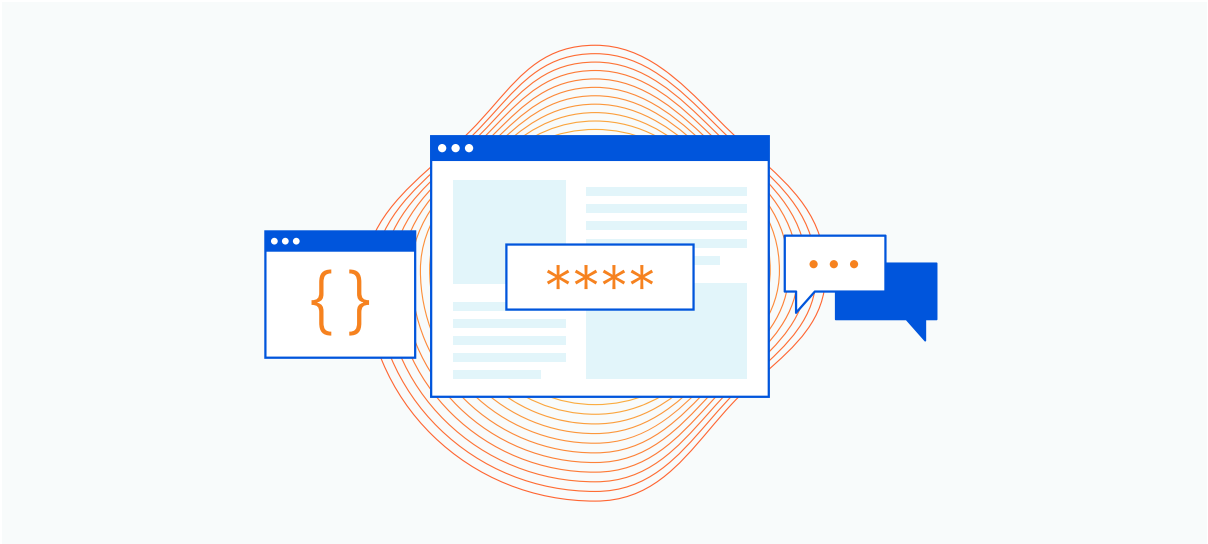
白皮书

CLOUDFLARE

API 安全指南

010 8524 1783 | enterprise@cloudflare.com | www.cloudflare.com

API 使（应用）世界正常运转



迄今为止，我们都知道，应用程序编程接口 (API) 使世界正常运转。更具体而言，API 使不同的现代应用程序之间能够相互通信。移动或 Web 应用程序可访问存储和处理数据的后端。API 可以是公共的，允许来自不同公司的应用程序进行通信；API 也可以是私有的，这是常见的做法，在这种情况下，用于集成内部应用程序以达成业务目标。

结果？更强大、成熟的应用程序、网站和移动应用，具有更广泛的功能和更多样化的数据。

例如，拼车公司可以通过支付公司的 API 添加支付服务，而非从头开始创建自己的支付服务。另一个例子是航班信息聚合网站。为了向我们显示航班时间、价格、目的地和我们想知道的所有其他机票信息，这些网站通过 API 调用连接到航空公司数据库，以提取正确的数据，并在我们的聚合搜索结果页面中显示出来。

API 的重要性不断提升，越来越多公司正将自己描述为“API 优先”。在某些情况下，一家公司的实际产品就是一个 API，商业模式以 API 的使用为中心。例如，如果一家提供天气数据的公司将他们的 API 作为产品，其他需要天气信息的公司将按月向他们支付 API 访问费用。在许多其他情况下，考虑到应用程序必须与其他应用程序交互的需求，API 是与交付实际产品功能的代码一起甚至在代码生成之前设计的——而不是在开发结束时绑定。

公司会投入时间和努力来精心设计他们的 API 策略，以考虑如何生成正确的数据，从而为商业模式正常运转及盈利奠定基础。

然而，构建完美的 API 并非易事，因为就像任何软件一样，漏洞无法避免，从而导致我们将在本文中探讨的安全挑战。

可以说，API 已经无处不在，而且未来几年内只会继续增长，它们必须受到保护。为此，我们将深入研究在考虑组织的 API 安全性时，API 攻击和防御的各个方面。

API 发展趋势（数据）

50%

Cloudflare 网络流量中 API 流量所占比例

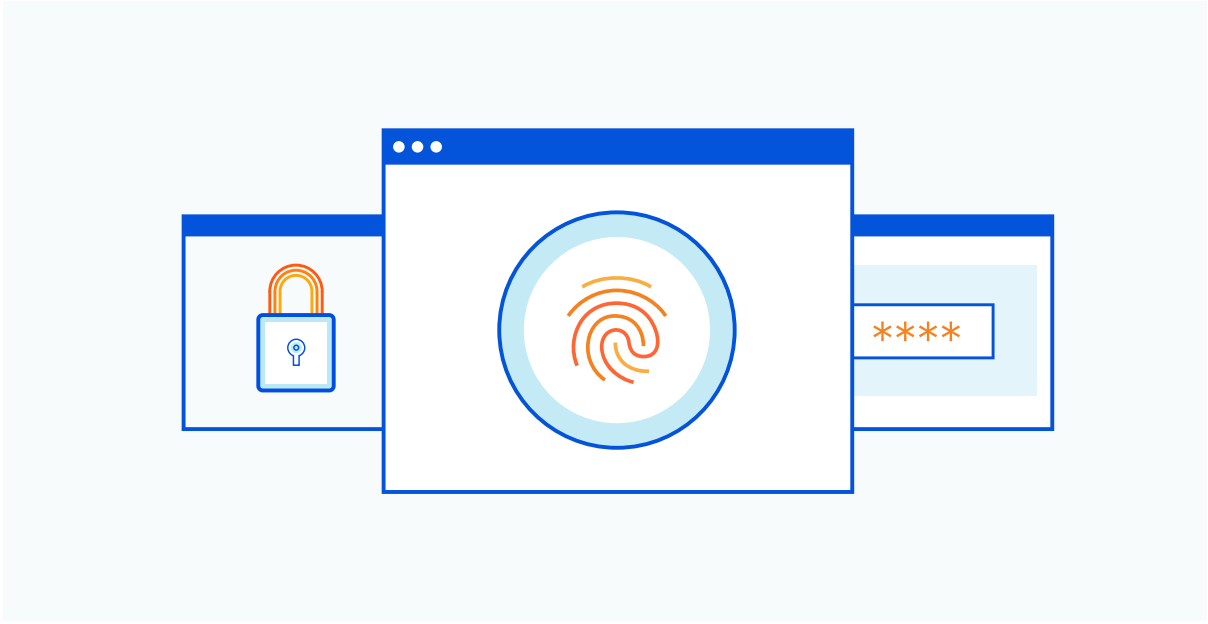
61%

API 流量年增长率

Programmable Web[1] 网站数据显示超过 2.4 万个已发布的知名 API。然而，大多数 API 是私有的，用于将内部应用程序连接起来。私有 API 的数量据估计达到数百万。

当我们说 API 正在加速增长时，Cloudflare 就是这种发展势头的直接见证者。根据来自 2021 年上半年的 Cloudflare Radar 数据， Cloudflare 网络上大约一半流量与 API 有关。此外，2020 年到 2021 年，API 流量增长了 61%。

鉴于 API 会暴露重要数据，我们开始发现 API 成为一个新的巨大攻击面，必须加以保护。我们是如何知道这一切的呢？近年来发生了多次针对 API 的知名攻击。



[1]https://www.programmableweb.com/apis/directory

不断扩大的攻击面

我们现在知道，API 无处不在，并且是现代商业成功的基础。它们公开应用程序逻辑，并与其他应用共享敏感数据。

然而，大家都毫不意外的是，攻击者也知道这一点，并打算充分利用企业这一不断扩大的攻击面。

也许 Gartner 的说法[2]是对的：到 2022 年，API 滥用将从“一种不常见的攻击方式变成一种最常见的攻击方式，导致企业 Web 应用程序的数据泄露。”

API 是否会成为最常见的攻击目标有待观察，但显然 API 将继续出现在攻击者的目标当中。

我们之所以说“继续”，是因为世界上已经发生了一些因 API 安全性较弱或 API 开发不考虑安全性而引起的高调入侵事件。

T-Mobile 在 2017 年成为 API 攻击的受害者，1500 万名购买新设备或申请 T-Mobile 帐户的用户的信息被泄露。数据包括姓名、地址、出生日期、社会安全号码、驾照号码和护照号码。据 Vice 报告，攻击是通过调整 API 调用中的电话号码参数进行的。任何用户的电话号码都可以查询，T-Mobile API 会发送回复，包括被查询人的敏感帐户数据。

另一次 API 相关入侵的目标是美国邮政 (USPS)，一个支持执行包裹跟踪的 API 被发现缺乏基本授权机制。某个用户登录后，使用通配符搜索参数，即可查询任何其他用户的帐户信息，调出数据集的所有记录。这使美国邮政 6000 万帐户持有者的数据陷入危险境地。

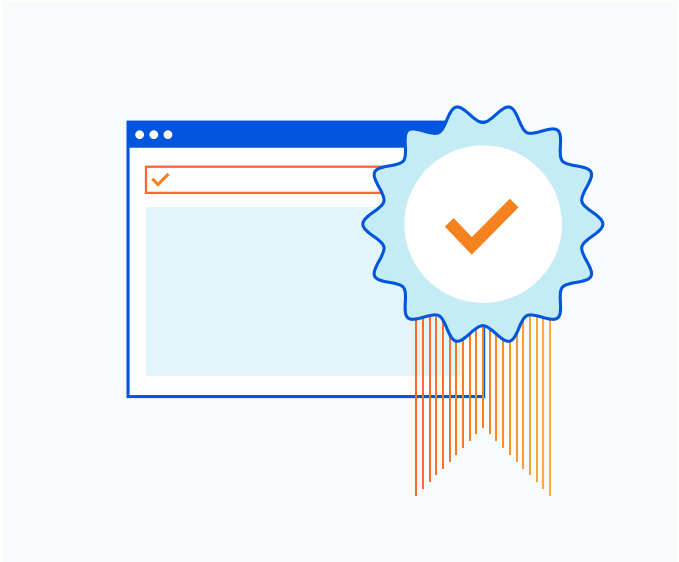
在 2019 年，印度一家大型本地搜索引擎 JustDial 泄露了所有客户数据，原因是他们暴露了已被新版本取代的旧版本 API。雪上加霜的是——系统实际上没有进行身份验证，这意味着任何人都可以调用 API 并从生产服务器获取数据。换句话说，访问用户数据不需要高级技术。

尽管 Facebook 在 GraphQL 方面处于领先地位，但他们的 API 也多次遭到入侵。例如，在 2019 年底，Facebook 的一个数据库被窃取，超过 2.6 亿用户的姓名、电话号码和用户 id 被置于风险之中。

许多组织都在保护 API 方面遇到麻烦。原因有很多。首先，不安全设计导致的底层 API 漏洞为攻击者打开了大门。更重要的是，到目前为止，组织还没有 API 优先的安全工具。也许他们会使用网络安全工具，如 WAF 或速率限制，但这些工具是用来保护应用程序的，而不是 API。这可能导致虚报等挑战，并清楚地表明，需要为主要是自动化的流量而设计的 API 安全性。

[2]来源：Gartner：《API 安全：如何保护您的 API》(“API Security: What You Need to Do to Protect Your APIs”)，2021 年 3 月

重制! 最新 OWASP 十大 API 风险公布



作为一个长期致力于提高应用程序安全性的组织，OWASP 基金会的参与给人们带来一丝光明。OWASP 以其十大 Web 应用安全风险而闻名，现已发布了十大 API 安全风险和漏洞清单。

事实是，长期以来让我们担心应用程序安全性的问题，也适用于构建和保护 API。

首先，任何 API 优先的组织在设计 API 时都必须首先考虑安全性。让我们研究一下刚才提到的一些攻击，以及它们利用的 OWASP 安全风险。

OWASP 十大 API 安全风险

1. 失效的对象级授权
2. 用户身份验证失效
3. 过度数据暴露
4. 缺乏资源与速率限制
5. 失效的函数级授权
6. 批量分配
7. 安全配置错误
8. 注入
9. 资产管理不当
10. 日志记录和监控不足

关键 API 安全挑战

1. 无效的身份验证和授权

让我们仔细研究一下上述攻击利用的几个关键 OWASP API 风险，首先是身份验证和授权。

JustDial 在其 API 端点上的身份验证是无效的，允许任何人向其发起调用。当实施身份验证时，只有具有正确的 TLS 证书、API 密钥、Web 令牌等的 API 调用才被允许发出请求，从而显著降低 API 的安全风险。

在 OWASP 清单上位居第一，很多 API 攻击都会利用薄弱、无效或不存在的授权，例如美国邮政和 T-Mobile 所遭受的攻击。无效的对象级授权很常见。攻击者会将获授权访问的对象的 ID 替换成未授权访问的另一个对象的 ID，从而利用 API 端点。通常情况下，只要更改某个请求中的对象 ID，就能获得对数据的未授权访问。

API 路径和查询参数包括正在访问的资源 ID：

授权调用：

`GET api.greatsampleapis.com/v1/users/235` 其中 235 是用户 ID。

被操纵的 API 调用可以通过改变对象标识符来获得未授权访问，在本例中，将 userID 从 235 改为 236，就能访问用户 236 的数据。

`GET api.greatsampleapis.com/v1/users/236`

如果没有实施授权控制，这一尝试就会成功。开发人员应该对他们的端点进行威胁建模，以确保攻击不能通过调整或修改对象的 ID 值来获得对其他数据的访问权。使用不可预测的对象 ID 值也有帮助，这样它们就不是连续的，不容易被猜中。

2. 批量分配、数据暴露和注入攻击

另一类攻击在响应中暴露太多数据，或者允许使用输入修改内部对象。

过度的数据暴露是指 API 试图广泛地公开对象属性并在响应中返回太多数据，依赖于客户端发出过滤数据的请求。

攻击者可以通过利用响应中的额外细节来创建更强大的攻击或钓鱼电子邮件。例如，如果一个响应返回所有以下数据，它就可以通过钓鱼电子邮件成功实施攻击：

```
{
 “Id”:213,
 “FirstName”:“Sanjay”,
 “LastName”:“Smythe”,
 “EmailAddress”:“ssmythe@hacketyhack.com”,
 “Occupation”:“Assistant to the Deputy Associate Vice Sub-undersecretary”,
 “DOB”:“1986-05-21”,
 “Bank”:“Easygo Financial”,
 “AccountNumber”:1362886306,
 “PetName”:“Aloysius”,
}
```

批量分配攻击允许 API 调用在 API 公开内部对象和变量时修改或利用内部值。

OWASP 对此有如下表述：

> “软件框架有时允许开发人员自动将 HTTP 请求参数绑定到程序代码变量或对象中，以使开发人员更容易使用该框架……攻击者有时可以使用这种方法来创建开发者从未想过的新参数，从而在程序代码中创建或覆盖非预期的新变量或对象。”

开发人员应该怎么做？他们应该理解在开发中调用批量分配时的潜在风险，并避免暴露任何可能被利用的内部对象或变量。此外，也应该考虑可被客户端更新的 Allowlisting 属性。

Web 应用程序长期以来都容易遭到注入攻击，API 也不例外。考虑到注入攻击已众所周知，我们不会详细讨论，说明一点就足够了：传递输入值前应该进行验证和清理。应该努力在 API 响应中使用数据泄漏预防，并限制可以返回的记录数量，以防止发生大规模泄露事件。

3. 资源滥用和影子/流氓 API

其他攻击会滥用 API，从而消耗过多计算资源，在类似拒绝服务 (DoS) 的攻击下变得不堪重负。如果不限制每个客户端/资源的请求数量、单个响应中返回的记录或请求负载大小等，就会为此类攻击敞开大门。

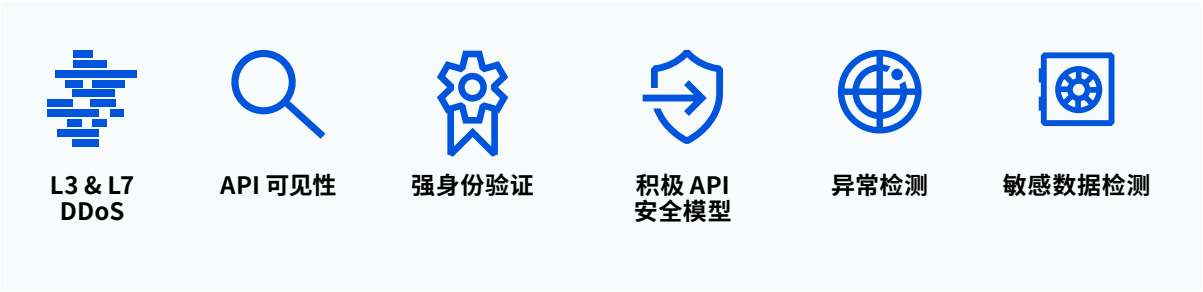
正如我们在 JustDial 攻击中看到的那样，生产 API 可能会被遗忘，从而成为影子或流氓 API，因为它们可能不受保护并被利用。与其他安全性一样，我们必须对 IT 资产或攻击面具有可见性，然后应用适当的安全性控制。对我们全部 API 端点的可见性也没有什么不同。

在开发 API 时，团队应该有一个精密的流程来跟踪 API 版本，以了解哪些 API 正在生产中，哪些 API 已被弃用。

保护 API 的注意事项

我们已经讨论了 API 是什么，为什么它们很重要，以及针对 API 的一般攻击。现在让我们来看看 Cloudflare 如何构建 API 安全性，以便保护 API 免受最常见的攻击。有效的 API 安全性必须考虑到方方面面的问题，包括可见性，积极的安全模型，阻止滥用，以及数据保护等。

Cloudflare API Shield



可见性的基础

可见性

就像安全的其他方面一样，要保护某样东西，我们必须看到它。API 也没有什么不同，尤其是当公司拥有数百甚至数千个 API 端点时。

API 发现和可见性是管理 API 的一个关键基础，以便组织拥有其 API 端点资产的清晰快照，以防止影子或流氓 API 造成问题。

正如我们从 JustDial 案例所见的那样，如果组织不能严密跟踪其 API，就有可能导致数据泄露。

API 纵深防御

API L7 保护

我们很早以前就已经部署了 Web 应用程序防火墙，保护应用程序以防第 7 层 DDoS 攻击。API 保护应该从许多可靠的控制开始，例如速率限制和 DDoS 保护，以防止拒绝服务攻击和暴力登录尝试，以及特定 IP 地址的一般滥用。这样将加强 API 的使用限制，并确保对抗 OWASP API 4 的可用性——缺乏资源和速率限制。

WAF 规则也应该用于识别和阻止针对 API 的常见攻击。

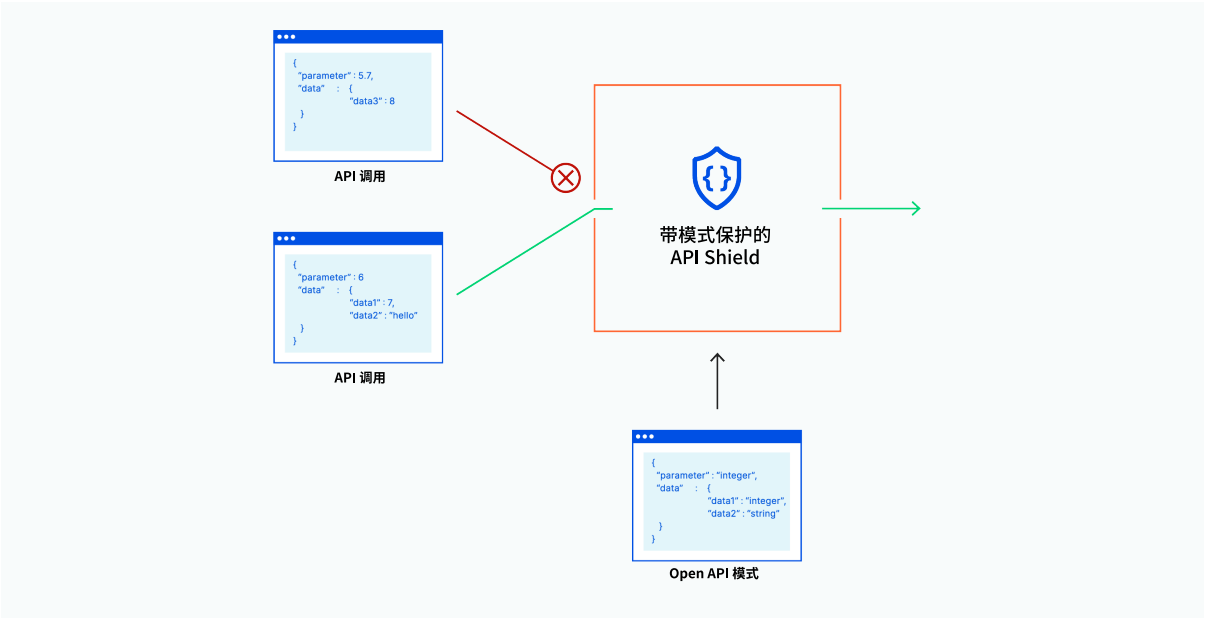
身份验证和授权

mTLS 身份验证

我们从 API 攻击中看到，缺乏身份验证可能带来灾难性的后果。身份验证必须从一开始就内置，并且应该使用双向 TLS 来加强，以便在移动或物联网等用例中实施基于证书的身份验证。这种方法是一种更积极的允许列表模型，它只允许来自具有有效证书的合法客户端的请求进行连接。

暴露凭据检查

API 也不能幸免于凭据填充攻击，这种攻击会循环使用窃取的凭据进行登录尝试。这些帐户凭据可能会受到组织控制之外的第三方入侵的影响。作为身份验证检查的一部分，API 安全性应该能够根据泄露凭据的数据库，在登录时扫描身份验证凭据。如果凭据确实已遭到泄露，API 安全性应该触发额外的安全措施，如密码重置或多因素身份验证，当然还会阻止这种尝试。



模式验证根据 API 模式对每个请求进行评估，记录或阻止不符合的请求。

积极的 API 保护

API 模式验证

开发人员会不遗余力地创建 API 模式，这个文档（或称基本规则）规定了期望他人与 API 交互的方式。API 模式会规定每个端点上的请求方法和操作 (GET /users, POST /users) 或每个操作的输入和输出参数。OpenAPI v3，也被称为 Swagger 标准，是定义 API 的最著名模式。

可靠的 API 安全性应该使用一个积极的 Zero Trust 模型，在该模式上强制执行。

有了模式后，就应该根据模式自动验证请求。除了那些经过验证符合模式的操作外，所有 API 操作都将被阻止。